

第4章 Redis持久化

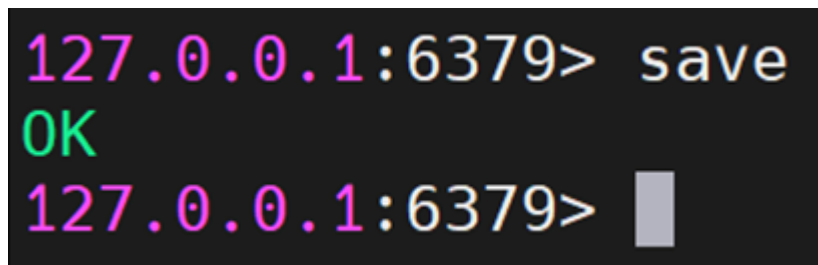
Redis是一个内存数据库，所以其运行效率非常高。但也存在一个问题：内存中的数据是不持久的，若主机宕机或Redis关机重启，则内存中的数据全部丢失。当然，这是不允许的。Redis具有持久化功能，其会按照设置以快照或操作日志的形式将数据持久化到磁盘。

根据持久化使用技术的不同，Redis的持久化分为两种：RDB与AOF。

1 RDB持久化

RDB, Redis DataBase, 是指将内存中某一时刻的数据快照全量写入到指定的rdb文件的持久化技术。RDB持久化默认是开启的。当Redis启动时会自动读取RDB快照文件，将数据从硬盘载入到内存，以恢复Redis关机前的数据库状态。

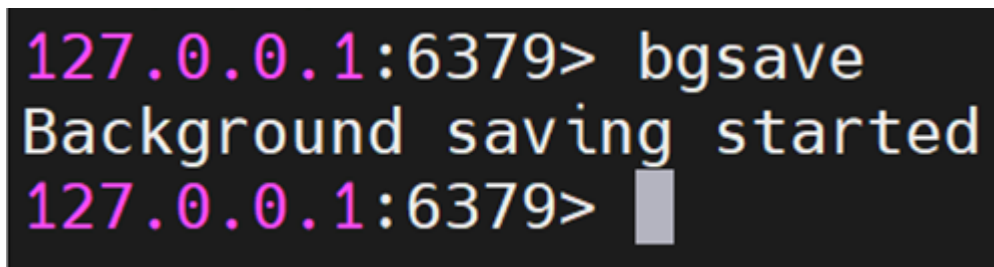
手动save命令



```
127.0.0.1:6379> save
OK
127.0.0.1:6379> █
```

通过在redis-cli客户端中执行save命令可立即进行一次持久化保存。save命令在执行期间会阻塞redis-server进程，直至持久化过程完毕。而在redis-server进程阻塞期间，Redis不能处理任何读写请求，无法对外提供服务。

手动bgsave命令



```
127.0.0.1:6379> bgsave
Background saving started
127.0.0.1:6379> █
```

通过在redis-cli客户端中执行bgsave命令可立即进行一次持久化保存。不同于save命令的是，正如该命令的名称一样，background save，后台运行save。bgsave命令会使服务器进程redis-server生成一个子进程，由该子进程负责完成保存过程。在子进程进行保存过程中，不会阻塞redis-server进程对客户端读写请求的处理。

自动条件触发

自动条件触发的本质仍是bgsave命令的执行。只不过是用户通过在配置文件中做相应的设置后，Redis会根据设置信息自动调用bgsave命令执行。

```
# Save the DB to disk.
#
# save <seconds> <changes> [<seconds> <changes> ...]
#
# Redis will save the DB if the given number of seconds elapsed and it
# surpassed the given number of write operations against the DB.
#
# Snapshotting can be completely disabled with a single empty string argument
# as in following example:
#
# save ""
#
# Unless specified otherwise, by default Redis will save the DB:
# * After 3600 seconds (an hour) if at least 1 change was performed
# * After 300 seconds (5 minutes) if at least 100 changes were performed
# * After 60 seconds if at least 10000 changes were performed
#
# You can set these explicitly by uncommenting the following line.
#
# save 3600 1 300 100 60 10000
```

该配置用于设置快照的自动保存触发条件，即save point，保存点。该触发条件是在指定时间段内发生了指定次数的写操作。除非另有规定，默认情况下持久化条件为save 3600 1 300 100 60 10000。其等价于以下三条：

- save 3600 1 # 在3600秒(1小时)内发生1次写操作
- save 300 100 # 在300秒(5分钟)内发生100次写操作
- save 60 10000 # 在60秒(1分钟)内发生1万次写操作

如果不启用RDB持久化，只需设置save的参数为空串即可：save “”。

持久化文件目录

```
# The working directory.
#
# The DB will be written inside this directory, with the filename specified
# above using the 'dbfilename' configuration directive.
#
# The Append Only File will also be created inside this directory.
#
# Note that you must specify a directory here, not a file name.
dir ./
```

配置文件中的dir属性用于指定RDB与AOF文件的生成目录。默认为Redis安装根目录。

2 AOF持久化

AOF, Append Only File, 是指Redis将每一次的写操作都以日志的形式记录到一个AOF文件中的持久化技术。当需要恢复内存数据时，将这些写操作重新执行一次，便会恢复到之前的内存数据状态。

AOF基础配置

AOF的开启

```
#  
# Please check https://redis.io/topics/persistence for more information.  
appendonly yes
```

默认情况下AOF持久化是没有开启的，通过修改配置文件中的appendonly属性为yes可以开启。

文件名配置

```
# For example, if appendfilename is set to appendonly.aof, the following file  
# names could be derived:  
#  
# - appendonly.aof.1.base.rdb as a base file.  
# - appendonly.aof.1.incr.aof, appendonly.aof.2.incr.aof as incremental files.  
# - appendonly.aof.manifest as a manifest file.  
appendfilename "appendonly.aof"
```

Redis 7在这里发生了重大变化。原来只有一个appendonly.aof文件，现在具有了三类多个文件：

- 基本文件：可以是RDB格式也可以是AOF格式。其存放的内容是由RDB转为AOF当时内存的快照数据。该文件可以有多个。
- 增量文件：以操作日志形式记录转为AOF后的写入操作。该文件可以有多个。
- 清单文件：用于维护AOF文件的创建顺序，保障激活时的应用顺序。该文件只有一个。

混合式持久化开启

```
# Redis can create append-only base files in either RDB or AOF formats. Using  
# the RDB format is always faster and more efficient, and disabling it is only  
# supported for backward compatibility purposes.  
aof-use-rdb-preamble yes
```

对于基本文件可以是RDB格式也可以是AOF格式。通过aof-use-rdb-preamble属性可以选择。其默认值为yes，即默认AOF持久化的基本文件为rdb格式文件，也就是默认采用混合式持久化。

AOF文件目录配置

```
# For convenience, Redis stores all persistent append-only files in a dedicated  
# directory. The name of the directory is determined by the appenddirname  
# configuration parameter.  
appenddirname "appendonlydir"
```

为了方便管理，可以专门为AOF持久化文件指定存放目录。目录名由appenddirname属性指定，存放在redis.conf配置文件的dir属性指定的目录，默认为Redis安装目录。

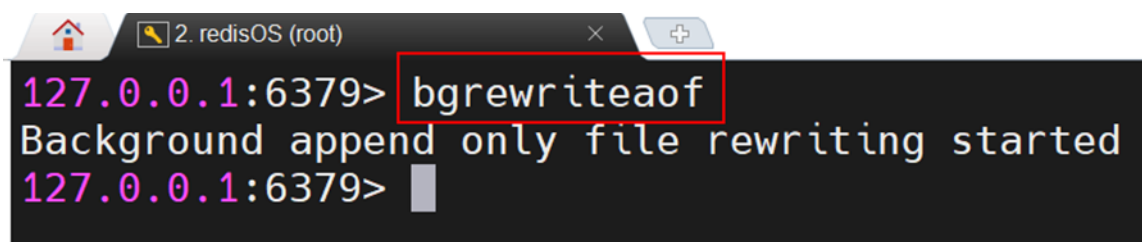
Rewrite机制

随着使用时间的推移，AOF文件会越来越大。为了防止AOF文件由于太大而占用大量的磁盘空间，降低性能，Redis引入了Rewrite机制来对AOF文件进行压缩。

手动开启rewrite

Rewrite过程的执行有两种方式。一种是通过bgrewriteaof命令手动开启，一种是通过设置条件自动开启。

以下是手动开启方式：



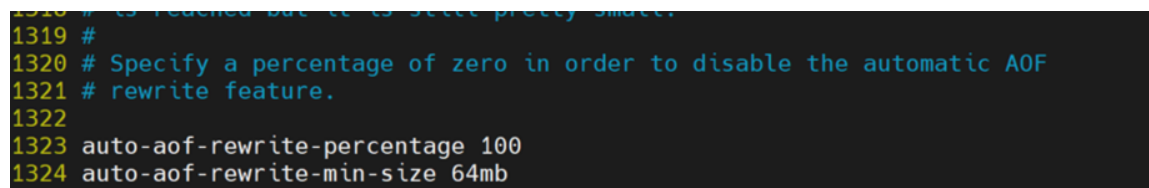
```
127.0.0.1:6379> bgrewriteaof
Background append only file rewriting started
127.0.0.1:6379>
```

该命令会使主进程redis-server创建出一个子进程bgrewriteaof，由该子进程完成rewrite过程。而在rewrite期间，redis-server仍是可以对外提供读写服务的。

自动开启rewrite

手动方式需要人办干预，所以一般采用自动方式。由于Rewrite过程是一个计算过程，需要消耗大量系统资源，会降低系统性能。所以，Rewrite过程并不是随时随地任意开启的，而是通过设置一些条件，当满足条件后才会启动，以降低对性能的影响。

下面是配置文件中对于Rewrite自动启动条件的设置。



```
1319 #
1320 # Specify a percentage of zero in order to disable the automatic AOF
1321 # rewrite feature.
1322
1323 auto-aof-rewrite-percentage 100
1324 auto-aof-rewrite-min-size 64mb
```

- auto-aof-rewrite-percentage：开启rewrite的增大比例，默认100%。指定为0，表示禁用自动rewrite。
- auto-aof-rewrite-min-size：开启rewrite的AOF文件最小值，默认64M。该值的设置主要是为了防止小AOF文件被rewrite，从而导致性能下降。

自动重写AOF文件。当AOF日志文件大小增长到指定的百分比时，Redis主进程redis-server会fork出一个子进程bgrewriteaof来完成rewrite过程。

其工作原理如下：Redis会记住最新rewrite后的AOF文件大小作为基本大小，如果从主机启动后就没有发生过重写，则基本大小就使用启动时AOF的大小。

如果当前AOF文件大于基本大小的配置文件中指定的百分比阈值，且当前AOF文件大于配置文件中指定的最小阈值，则会触发rewrite。

3 RDB与AOF对比

RDB优势与不足

RDB优势

- RDB文件较小
- 数据恢复较快

RDB不足

- 数据安全性较差
- RDB文件可读性较差

AOF优势与不足

AOF优势

- 数据安全性高
- AOF文件可读性强

AOF不足

- AOF文件较大
- 写操作会影响性能
- 数据恢复较慢

持久化技术的选择

官方推荐采用RDB与AOF混合式持久化。但如果对数据安全性要求不高，则可以仅使用RDB方式。不过，不建议单独使用AOF方式。

如果Redis仅仅就是作为整个系统的缓存，则无需使用任何持久化技术。

